UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/829,668	04/22/2004	Douglas C. Burger	119832-163869	6831

60172 7590 03/09/2010

SCHWABE, WILLIAMSON & WYATT, P.C.
1420 FIFTH, SUITE 3010
SEATTLE, WA 98101

EXAMINER	
FENNEMA, ROBERT E	

ART UNIT	PAPER NUMBER
2183	

MAIL DATE	DELIVERY MODE
03/09/2010	PAPER

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

Office Action Summary	Application No.	Applicant(s)	
	10/829,668	BURGER ET AL.	
	Examiner	Art Unit	
	Robert Fennema	2183	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *04 January 2010*.

2a)☐ This action is **FINAL**. 2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *37-44,46-50,52-58 and 60* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *37-44,46-50,52-58 and 60* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☒ Information Disclosure Statement(s) (PTO/SB/08) Paper No(s)/Mail Date *1/4/2010*.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____.
5) ☐ Notice of Informal Patent Application
6) ☐ Other: _____.

DETAILED ACTION

1. Claims 37-44, 46-50, 52-58, and 60 are pending. Claims 37-38, 47, 52, 56-57, and 60 amended as per Applicant's request. Claim 59 cancelled as per Applicant's request.

2. Examiner notes the entry of the following papers:

- Amendment to the claims and remarks filed 1/4/2010
- IDS filed 1/4/2010

3. A request for continued examination under 37 CFR 1.114, including the fee set forth in 37 CFR 1.17(e), was filed in this application after final rejection. Since this application is eligible for continued examination under 37 CFR 1.114, and the fee set forth in 37 CFR 1.17(e) has been timely paid, the finality of the previous Office action has been withdrawn pursuant to 37 CFR 1.114. Applicant's submission filed on 1/4/2010 has been entered.

Claim Objections

4. In Claim 37, Line 7, "the associate operands" should read "the associated operands".

5. In Claim 37, Lines 7-8, Claim 47, Line 12, Claim 57, Line 11, and Claim 60, Line 24, "prior to the associated operands of the subset of instructions are available" does

not make sense. Examiner is interpreting it as "prior to the availability of the associated operands of the subset of instructions", however, correction or clarification is required.

6. In Claim 37, Line 9, "associated operands" should read "the associated operands" for proper antecedent basis.

7. In Claim 37, Line 6, Claim 43, Lines 2-3 and 5-6, and Claim 44, Lines 5-6, "the subset of interconnected preselected computation nodes" should read either "the subset of interconnected computation nodes" or "the preselected subset of interconnected computation nodes" for proper antecedent basis.

8. In Claim 37, Lines 9-13, "the preselected computation nodes" lacks antecedent basis. Applicant has defined both "a subset of interconnected computation nodes", which are preselected from "a plurality of interconnected computation nodes". Examiner will interpret it as "the subset of the plurality of interconnected preselected computation nodes" to be consistent with the rest of the claim.

9. In Claim 37, Lines 10-11, "a first of the preselected..." and "a second of the preselected"... should read "a first node of the preselected" and "a second node of the preselected..." to be consistent with the remainder of the claims.

10. In Claim 37, Line 16, "receives associated operands" should read "receives the associated operands" for proper antecedent basis.

11. In Claim 38, Line 9, "the first operand" should read "the first associated operand".

12. In Claim 47, Line 7, "a group of instructions" should read "the group of instructions".

13. In Claim 47, Line 8, and Claim 60, Lines 15, 22, 29, and 30, "preselected" should be removed from the claim for proper antecedent basis.

14. In Claim 47, Line 15, "wherein receive associated operands" does not make grammatical sense, and should read "wherein the receiving of the associated operands for execution".

15. In Claim 47, Line 15, "a first of the..." should read "a first computation node of the..." to avoid antecedent basis issues with the remainder of the claims.

16. In Claim 47, Line 15, "computation nodes" should read "interconnected computation nodes".

17. In Claim 47, Line 17, "a second of the..." should read "a second computation node of the..." to avoid antecedent basis issues with the remainder of the claims.

18. In Claim 53, Line 2, Claim 54, Line 2, there is an unnecessary space between instructions and the comma.

19. In Claims 53-55, "Lines 2, "if executed, enable" should read "if executed, to enable" to be grammatically correct.

20. In Claim 57, Line 3, "storage medium" should read "a storage medium".

21. In Claim 57, Line 3, "the processor" should read "a processor".

22. In Claim 57, Line 11, "the subset of the second instruction" should read "the subset of the second instructions".

23. In Claim 57, Lines 13-14, "the preselected computation nodes" should read "the interconnected computation nodes".

24. In Claim 60, Line 3, "the group" should read "the group of instructions".

25. In Claim 60, Line 5, "wherein individual of the interconnected computing nodes" does not make sense (and should also read "wherein individual of the plurality of interconnected computing nodes". Examiner believes the claim is intended to read either "wherein each individual interconnected computing node" or "wherein individual interconnected computing nodes". Examiner is interpreting the claim as the former, but correction is required. This error remains from the last action, and if correction is not made in the next response, Examiner will hold the response non-compliant.

26. In Claim 60, Line 6, "computing resource" should read "a computing resource".

27. In Claim 60, Line 8, "interconnect resource" should read "an interconnect resource.

Claim Rejections - 35 USC § 101

28. 35 U.S.C. 101 reads as follows:

> Whoever invents or discovers any new and useful process, machine, manufacture, or composition of matter, or any new and useful improvement thereof, may obtain a patent therefor, subject to the conditions and requirements of this title.

Claims 47-50 and 52-55 are rejected under 35 U.S.C. 101 because the claimed invention is directed to non-statutory subject matter. The claims recite a machine-accessible medium, which has not been specifically defined in the specification, only as a combination of the definitions provided on Page 14 of the specification, which includes both statutory storage mediums, and non-statutory communications mediums. To overcome this rejection, Applicant can either amend the claims to recite a computer

readable storage medium, instead of a machine-accessible medium, or to recite a "non-transitory machine-accessible medium", which excludes all non-statutory embodiments such as carrier waves or signals.

Claim Rejections - 35 USC § 103

29. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

30. Claims 37-39, 43-44, 46-48, 50, 52-58, and 60 are rejected under 35 U.S.C. 103(a) as being unpatentable over Requa et al. ("The Piecewise Data Flow Architecture: Architectural Concepts", herein Requa), in view of Patterson et al. ("Computer Architecture, A Quantitative Approach", herein Patterson).

31. As per Claim 37, Requa teaches: A method, comprising:

assigning a group of instructions selected from the plurality of groups of instructions partitioned from a program to a subset of interconnected computation nodes preselected from a plurality of interconnected computation nodes (Page 426, first column, second paragraph, the blocks are sent to processors), the instructions having associated operands (Page 427, for instructions to have dependencies, they must have operands); and

executing the subset of instructions as each one of the instructions in the subset of instructions loaded into the frame of buffers receives associated operands for execution (Page 433, first column, third paragraph, an instruction waits for input operands, then is executed. Also see Page 435, Instruction Issue section. When the instruction in the list has all required operands, it is sent to the instruction-issue section, and then a processor), but fails to teach:

receiving associated operands of the subset of instructions by the preselected computation nodes, including a first of the preselected computation nodes directly receiving a first associated operand of a first instruction loaded into the first preselected computation node, from a second of the preselected computation nodes, wherein the first computation node has an input port capable of being coupled to the second computation node to enable the directly receiving of the first associated operand; and

loading a subset of instructions of the assigned group of instructions into a frame of buffers comprising stores disposed on the subset of interconnected preselected computation nodes having been assigned the group of instructions, prior to the associate operands of the subset of instructions are available.

Requa teaches loading instructions into a centralized instruction buffer, and only when all operands are available for execution, is it sent to the appropriate processor for execution, thus, Requa does not teach a buffer for each node, receiving instructions prior to the operands for said instructions being present. However, Patterson teaches of two methods to allow for dynamic scheduling, which has the advantage of removing multiple hazards: scoreboards and reservation stations. Requa's system is essentially a

scoreboard, a centralized buffer that keeps track of when instruction operands are available in the register file, then issues when appropriate. However, Patterson also suggests a reservation station, which has additional advantages in that it removes additional WAW and WAR hazards (Page 252), by using register renaming, and furthermore, distributes the logic by having a reservation station in each functional unit, instead of in one centralized location (Page 252), and further does not require data to be written into the register file before it can be accessed by the instruction. The advantages of the distribution are explained on Page 257 of Patterson, where by having each functional unit (processor) having its own reservation station, when results are broadcast, multiple instructions can be issued at once, wherein a centralized issue buffer cannot do so, in addition to the other advantages listed above. Additionally, a reservation station system such as Patterson's allows each processor to directly send data to another processor, via the bus, without the data first being stored in the register file. Therefore, given the aforementioned advantages, one of ordinary skill in the art would have been motivated to implement reservation station logic in Requa's architecture, which would involve both distribution of instruction buffers into each functional unit, and the issuing of instructions to those buffers prior to the operands being available.

32. As per Claim 38, Requa teaches: The method of claim 37, further comprising storing the first associated operand in a first store of the first computation node, wherein

the first store is coupled to the input port (PDF, paragraph 1, input operands are stored in registers),

storing the first instruction in a second store of the first computation nodes, wherein the second store is coupled to an instruction sequencer (Page 427, one of the processor-specific FIFOs, or in Patterson, the reservation station for each node also holds instructions),

matching the first associated operand with the first instruction by an instruction wakeup unit (PDF, paragraph 1, the operand source fields are modified as data comes in),

executing the first instruction by an execution unit of the first computation nodes using at least the first operand to produce output data (PDF, paragraph 1, the instructions are executed after receiving inputs),

routing the output data to an output port of the first computation nodes, wherein the output port is capable of being coupled to a third of the preselected computation nodes (Figure 1 and Page 427, second column, paragraph 2. Any consumer can receive any data from the interconnection network, thus all processors are capable of sending data to any other processor) to directly provide the output data to the third preselected computation node (Patterson, Page 254, any processor can receive results directly from any other processor via the bus).

33. As per Claim 39, Requa teaches: The method of claim 37, wherein at least one of the plurality of groups of instructions is a basic block (Page 426, first column third

paragraph).

34. As per Claim 43, Patterson teaches: The method of claim 37, wherein loading the group of instructions into a frame of buffers comprising stores disposed on the subset of interconnected preselected computation nodes includes:

sending at least two instructions selected from the group of instructions from an instruction sequencer to a selected computation node included in the subset of interconnected preselected computation nodes for storage in a store of the selected computation node, prior to the at least two instructions having all necessary associated operands for execution (Page 252).

35. As per Claim 44, Requa teaches: The method of claim 37, wherein executing the subset of instructions loaded into the frame of buffers as each one of the instructions in the subset of instructions receives associated operands for execution includes:

matching at least one instruction selected from the group of instructions with at least one operand received from an other computation node included in the subset of interconnected preselected computation nodes (Page 427, second column, second paragraph, where any consumer (processor) can receive any data, and instructions waiting to execute wait for results from the processor before executing).

36. As per Claim 46, Requa teaches: The method of claim 37, further comprising concurrently assigning another group of instructions selected from the plurality of

groups of instructions to another or the same subset of interconnected preselected computation nodes for concurrent execution using one or more other frames of buffers comprising stores disposed on the another or same subset of interconnected preselected computation nodes (Page 436, Second Column, Second and Third paragraphs. Requa discusses that blocks can overlap each other as long as they do not need results from each other. Additionally, in the third paragraph, Requa discusses that the PDF architecture is capable of supporting multiple program executions simultaneously, which would also read on the limitation):

wherein the two groups of instructions are capable of concurrent execution (Page 436, Second Column, Second and Third paragraphs. Requa discusses that blocks can overlap each other as long as they do not need results from each other. Additionally, in the third paragraph, Requa discusses that the PDF architecture is capable of supporting multiple program executions simultaneously, which would also read on the limitation).

37. As per Claim 47, Requa teaches: An article comprising a machine-accessible medium having machine executable instructions stored therein, if executed, enable a machine to:

assign a subset of a group of instructions selected from a plurality of groups of instructions partitioned from a program, to a subset of interconnected computation nodes of the machine (Page 426, first column, second paragraph, the blocks are sent to processors), the instructions having associated operands (Page 427, for instructions to have dependencies, they must have operands); and

load the subset of a group of instructions into a frame of buffers (Page 426, first column, second paragraph, also see Page 433, "The PDF Block Processor". Specifically, see Page 434, Figure 9. The Instruction List, as it is used by all processors, and is a buffer, it is a buffer which spans all the connected nodes/processor),

wherein the loaded instructions are executed as each one of the instructions receives associated operands for execution (Page 433, first column, third paragraph, an instruction waits for input operands, then is executed. Also see Page 435, Instruction Issue section. When the instruction in the list has all required operands, it is sent to the instruction-issue section, and then a processor), but fails to teach:

a frame of buffers comprising stores disposed on the subset of interconnected preselected computation nodes, and

wherein receive associated operands includes at least a first of the subset of computation nodes directly receiving a first associated operand of a first instruction from a second of the subset of interconnected computation nodes coupled to the first computation node,

wherein the subset of the group of instructions is loaded into the frame of buffers prior to the associated operands of the subset of the group of instructions are available.

Requa teaches loading instructions into a centralized instruction buffer, and only when all operands are available for execution, is it sent to the appropriate processor for execution, thus, Requa does not teach a buffer for each node, receiving instructions prior to the operands for said instructions being present. However, Patterson teaches of two methods to allow for dynamic scheduling, which has the advantage of removing

multiple hazards: scoreboards and reservation stations. Requa's system is essentially a scoreboard, a centralized buffer that keeps track of when instruction operands are available in the register file, then issues when appropriate. However, Patterson also suggests a reservation station, which has additional advantages in that it removes additional WAW and WAR hazards (Page 252), by using register renaming, and furthermore, distributes the logic by having a reservation station in each functional unit, instead of in one centralized location (Page 252), and further does not require data to be written into the register file before it can be accessed by the instruction. The advantages of the distribution are explained on Page 257 of Patterson, where by having each functional unit (processor) having its own reservation station, when results are broadcast, multiple instructions can be issued at once, wherein a centralized issue buffer cannot do so, in addition to the other advantages listed above. Additionally, a reservation station system such as Patterson's allows each processor to directly send data to another processor, via the bus, without the data first being stored in the register file. Therefore, given the aforementioned advantages, one of ordinary skill in the art would have been motivated to implement reservation station logic in Requa's architecture, which would involve both distribution of instruction buffers into each functional unit, and the issuing of instructions to those buffers prior to the operands being available.

38. As per Claim 48, Requa teaches: The article of claim 47, wherein the machine executable instructions, if executed, further enable the machine to partition the program

into the plurality of groups of instructions during compilation of the program (Page 429, first column, "PDF Architecture").

39. As per Claim 50, Requa teaches: The article of claim 47, wherein the machine-accessible medium further includes instructions, if executed, enable the machine to:

statically assign each of the plurality of groups of instructions to a subset of interconnected preselected computation nodes preselected from a plurality of interconnected computation nodes for execution (Page 432, see Figure 8, and column 1, paragraph 2).

40. As per Claim 52, Requa teaches: The article of claim 47, wherein the machine-accessible medium further includes instructions, if executed, enable the machine to:

generate a wakeup token to reserve an output data channel of the second computation node to directly route the first associated operand from the second computation node to the first computation node (PDF, Page 427, section column, second paragraph, also see Patterson, Page 254).

41. As per Claim 53, Requa teaches: The article of claim 47, wherein the machine-accessible medium further includes instructions, if executed, enable the machine:

to repeat said loading until the entire group of instructions are executed (Page 433, in order to execute a block, this clearly has to occur), and

to detect execution termination of the group of instructions including an output having architecturally visible data (Page 433, second column first paragraph, a flag is set when an execution is done, also see Page 430, second paragraph); and

committing the architecturally visible data to a register file (Page 430, second paragraph).

42. As per Claim 54, Requa teaches: The article of claim 47, wherein the machine-accessible medium further includes instructions, if executed, enable the machine to repeat said loading until the entire group of instructions are executed (Page 433, in order to execute a block, this clearly has to occur), and to detect execution termination of the group of instructions including an output having architecturally visible data (Page 433, second column first paragraph, a flag is set when an execution is done, also see Page 430, second paragraph); and

committing the architecturally visible data to a memory (Page 430, second paragraph).

43. As per Claim 55, Requa teaches: The article of claim 47, wherein the machine-accessible medium further includes instructions, in executed, enable the machine to route an output datum arising from executing one of the subset of instructions to a consumer node included in the plurality of interconnected preselected computation nodes, wherein the address of the consumer node is included in a token associated with

at least one instruction included in the subset of instructions (Page 429, second column, second paragraph).

44. As per Claim 56, Requa teaches: The method of claim 37 further comprising repeating said loading and executing until the entire group of instructions have been executed (Page 433, in order to execute a block, this clearly has to occur).

45. As per Claim 57, Requa teaches: An apparatus, comprising:

a plurality of interconnected computation nodes (Page 427, Figure 1, the Scalar, Memory, and SIMD Processors); and

storage medium coupled to the processor and configured to have first instructions stored therein to be executed by the processor (Page 427, Main memory), wherein the first instructions are configured to:

assign a group of second instructions selected from a plurality of groups of second instructions partitioned from a program to a preselected subset of the plurality of interconnected computation nodes, the second instructions having associated operands (Page 426, first column, second paragraph, the blocks are sent to processors); and

wherein the subset of second instructions are executed as each one of the instructions in the subset of second instructions loaded into the frame of buffers receives associated operands for execution (Page 433, first column, third paragraph, an instruction waits for input operands, then is executed. Also see Page 435, Instruction

Issue section. When the instruction in the list has all required operands, it is sent to the instruction-issue section, and then a processor), but fails to teach:

wherein at least a first of the associated operands of a first of the second instructions loaded into a first of the preselected computation nodes is directly received form a second of the preselected computation nodes,

causing a subset of the second instructions of the assigned group of second instructions to be loaded into a frame of buffers comprising stores disposed on the subset of interconnected computation nodes having been assigned the group of second instructions, prior to the associated operands of the subset of the second instruction are available.

Requa teaches loading instructions into a centralized instruction buffer, and only when all operands are available for execution, is it sent to the appropriate processor for execution, thus, Requa does not teach a buffer for each node, receiving instructions prior to the operands for said instructions being present. However, Patterson teaches of two methods to allow for dynamic scheduling, which has the advantage of removing multiple hazards: scoreboards and reservation stations. Requa's system is essentially a scoreboard, a centralized buffer that keeps track of when instruction operands are available in the register file, then issues when appropriate. However, Patterson also suggests a reservation station, which has additional advantages in that it removes additional WAW and WAR hazards (Page 252), by using register renaming, and furthermore, distributes the logic by having a reservation station in each functional unit, instead of in one centralized location (Page 252), and further does not require data to

be written into the register file before it can be accessed by the instruction. The advantages of the distribution are explained on Page 257 of Patterson, where by having each functional unit (processor) having its own reservation station, when results are broadcast, multiple instructions can be issued at once, wherein a centralized issue buffer cannot do so, in addition to the other advantages listed above. Additionally, a reservation station system such as Patterson's allows each processor to directly send data to another processor, via the bus, without the data first being stored in the register file. Therefore, given the aforementioned advantages, one of ordinary skill in the art would have been motivated to implement reservation station logic in Requa's architecture, which would involve both distribution of instruction buffers into each functional unit, and the issuing of instructions to those buffers prior to the operands being available.

46. As per Claim 58, Requa teaches: The apparatus of claim 57, wherein at least one of the plurality of groups of second instructions is a selected of one a basic block, a hyperblock or a superblock (Page 426, first column third paragraph).

47. As per Claim 60, Requa teaches: A system, comprising:

a plurality of interconnected computing nodes configured to be pre-selectable to cooperatively execute a subset of a group of instructions (Page 427, Figure 1, the Scalar, Memory, or SIMD Processors), wherein the group is one of a plurality groups of instructions partitioned from a program (Page 426, blocks) and the instructions have

associated operands (Page 427, for instructions to have dependencies, they must have operands);

wherein individual of the interconnected computing nodes includes:

computing resource including an execution unit configured to execute instructions (Page 427, Figure 1, the Scalar, Memory, or SIMD Processor); and

interconnect resource coupled to the computing resource to enable the computing node to cooperate with the other interconnected computation nodes to execute the group of instructions by successively executing subgroups of the group of instructions (Page 427, Second Column, Second paragraph, all processors/nodes are connected to each other through an interconnection network. Page 433 discloses successive execution of parts of a block);

wherein the interconnect resource includes:

at least one input port capable of coupling the computing resource to at least a first other preselected computation node included in the plurality of interconnected preselected computation nodes, and the input port is configured to receive input data (Page 427, Second Column, Second paragraph, every node is connected to every other node, requiring an input port),

a first store coupled to the at least one input port, and configured to store the input data (Page 427, Second Column, Second paragraph, the FIFO queue),

a second store coupled to the execution unit, and configured to receive and store at least one instruction of a subgroup, the second store being a part of a frame of buffers spanning the plurality of interconnected preselected computation nodes to store

a subgroup of instructions loaded into the frame of buffers prior to the associated operands of the subgroup of instructions are available (Page 426, first column, second paragraph, also see Page 433, "The PDF Block Processor". Specifically, see Page 434, Figure 9. the Instruction List, as it is used by all processors, and is a buffer, it is a buffer which spans all the connected nodes/processor, and the instructions do not necessarily have their operands when first entered into the buffer),

an instruction wakeup unit to match the input data to the at least one stored instruction (PDF, paragraph 1, the operand source fields are modified as data comes in),

an output port coupled to the execution unit and capable of coupling the computing resource to a second other preselected computation node included in the plurality of interconnected preselected computation nodes (Page 427, Second Column, Second paragraph, every node is connected to every other node, requiring an output port), and

a router coupled to the execution unit, and configured to direct an output data of the execution unit to the output port for provision to the second other computation node (Page 427, the FIFO queue also acts as a kind of router), but fails to teach:

the input port, output port, and router directly coupling the resource to a node for receiving or sending data.

Requa does not teach that each node is directly connected to each other, in order to transfer data from one to the other, they must all transfer data into the instruction result registers (the register file) before being able to send data to another

processor. However, as is well known in the art, Patterson describes on Page 146 that this is what is known as a data hazard, and causes stalls or incorrect performance to occur. Patterson discloses that an easy and well known solution to this problem is data forwarding (Page 147-148), where instead of having to send data to the register file and into the functional units, one can just create a direct line path between the output of a unit and the input, then saving several clock cycles of waiting for the result. Given that data forwarding is well known in the art to avoid the problem of data hazards, one of ordinary skill in the art would have been motivated to allow for direct connections between the different nodes of Requa in order to overcome the deficiency in the architecture.

48. Claims 40-41 are rejected under 35 U.S.C. 103(a) as being unpatentable over Requa and Patterson, in view of Official Notice.

49. As per Claim 40, Requa teaches: The method of claim 37, but fails to teach: wherein at least one of the plurality of groups of instructions is a hyperblock.

Requa teaches of a system which uses basic blocks, but does not teach that the groups may be hyperblocks. However, Examiner is taking Official Notice that one of ordinary skill in the art would be capable and motivated to use hyperblocks in lieu of basic blocks, to take advantage of the ability to have multiple exits from a block (While Applicant has not provided a definition of a hyperblock, Examiner has found it to represent a block with one entrance and potentially (but not necessarily) more than one

exit).

50. As per Claim 41, Requa teaches: The method of claim 37, but fails to teach: wherein at least one of the plurality of groups of instructions is a superblock.

Requa teaches of a system which uses basic blocks, but does not teach that the groups may be superblocks. However, Examiner is taking Official Notice that one of ordinary skill in the art would be capable and motivated to use superblocks in lieu of basic blocks, to take advantage of the ability to have multiple exits from a block (While Applicant has not provided a definition of a superblock, Examiner has found it to represent a block with one entrance and potentially (but not necessarily) more than one exit, however, Examiner is unclear how a superblock is different from a hyperblock).

51. Claims 42 and 49 are rejected under 35 U.S.C. 103(a) as being unpatentable over Requa and Patterson, in view of Fisher.

52. As per Claim 42, Requa teaches: The method of claim 37, but fails to teach: wherein at least one of the plurality of groups of instructions is an instruction trace constructed by a hardware trace construction unit at run time.

While Requa teaches the method as disclosed in Claim 37, Requa does not teach about traces, or a trace construction unit to construct such a trace. However, Fisher teaches of Trace Scheduling, where the basic blocks used by Requa are compacted, and instead use traces (Page 462, Section D). The advantage to this

compaction method using traces allows for more efficient parallel code, done in a manner far more efficient than previous methods (Abstract). Given this advantage, one of ordinary skill in the art would have been motivated to use these traces in place of the basic blocks as taught by Requa to further increase the efficiency of the system.

53. As per Claim 49, Requa teaches: The article of claim 47, but fails to teach:

wherein the machine executable instructions, if executed, further enable the machine to partition the program into the plurality of groups of instructions is performed during run-time.

While Requa teaches the article as disclosed in Claim 47, Requa does not teach that the partitioning of the program is done by a trace mapper. However, Fisher teaches of Trace Scheduling, where the basic blocks used by Requa are compacted, and instead use traces (Page 462, Section D), created and optimized by a scheduler (Page 482, second column, second paragraph). The advantage to this compaction method using traces allows for more efficient parallel code, done in a manner far more efficient than previous methods (Abstract). Given this advantage, one of ordinary skill in the art would have been motivated to use these traces in place of the basic blocks as taught by Requa to further increase the efficiency of the system.

Response to Arguments

54. Applicant has argued on Pages 9-10 of the remarks that Examiner has admit that limitations in Claim 60 were not taught by Requa in other claims, however, Examiner

disagrees with this statement. Claim 60 recites a store being a part of a frame of buffers spanning the plurality of buffers, while the other claims recite a store disposed on each of the nodes. These are two very different limitations, one explicitly indicates that there is a different and distinct buffer for each node, and one indicates that it is part of a buffer that is connected to a larger buffer system. Therefore, Applicant's assertion is incorrect, because Claim 60 does not claim what the other independent claims claim. In regard to the new limitations added to Claim 60, Examiner refers to the rejection above for details, but notes that in general, it appears to be taught by the concept of forwarding data between two functional units to avoid having to indirectly get results from the register file, which is much slower than just passing the data directly.

55. Regarding Applicant's arguments with respect to the 103 rejections, Examiner notes that in order for a reservation station to get operands without using the register file, each reservation station monitors the common data bus, and as soon as a functional unit outputs data, if the reservation station needs it, it takes the data for itself, thus, there is a direct connection established between two different functional units, in the sense that the data output from one functional unit is input directly to another functional unit without any intermediate storage or buffering. Additionally, the teaching of forwarding would also appear to apply for these claims as well.

56. Additionally, Examiner strongly suggests re-wording the claims to simpler language, as all of the "first and seconds" of "a subset of interconnected nodes

preselected from a plurality of interconnected computation nodes" makes the claim very confusing, and has lead to a large number of antecedent basis issues, as the Applicant uses inconsistent language throughout the claims, and it is not always readily apparent if the Applicant is referring to the subset of nodes, the plurality of nodes, or a subset of the subset of nodes.

57. If Applicant has any questions about this action, Applicant is welcome to contact the Examiner at the phone number listed below to arrange for an interview.

Conclusion

Any inquiry concerning this communication or earlier communications from the examiner should be directed to Robert Fennema whose telephone number is (571)272-2748. The examiner can normally be reached on Monday-Thursday, 9:30-6:30.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Eddie Chan can be reached on (571) 272-4162. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free). If you would like assistance from a USPTO Customer Service Representative or access to the automated information system, call 800-786-9199 (IN USA OR CANADA) or 571-272-1000.

/Robert Fennema/
Examiner
Art Unit 2183